REMARKS

Introduction

Claims 2-5 have been cancelled. Claim 1 has been amended. Claims 1 and 6-16 remain in the application, of which claim 1 is in independent form.

Objection to the Drawings

The Examiner has indicated that the text in FIGs. 6-12 is too small to read. Upon review of the drawings, applicants note that the text in FIGs. 6-12 is large and readable. Applicants also note, however, that FIG. 5, which illustrates a logical mapping of FIGs. 6-12, includes small reference boxes representing FIGs. 6-12. These representations contain small text.

By this Amendment, FIG. 5 has been amended to remove the small text in the logical representations of FIGs. 6-12. No new matter has been added by way of this amendment. Accordingly, withdrawal of the objection to the drawings is requested.

Rejections under 35 U.S.C. § 103(a)

Claims 1-6, 9, 10 and 13 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,893,108 (*Srinvasan*) in view of Learn Java Now, Microsoft Press, 1996 (*Davis*).

Amended claim 1 is directed to a method for representing a relational database table as a object in an object-oriented operating system. The method includes the steps of "providing a reference to a primary key having a one-to-one mapping to a table entry in the relational database," "overloading the load method in the object-oriented operating system to load a latest instance of a table entry," and "overloading a save method in the object-oriented operating system to save an instance of a table entry." The method also includes "overloading a remove method in the object-oriented operating system to remove an instance of a table entry," wherein "overloading a remove method in the object-oriented operating system removes itself and any child instances," "overloading a save method in the object-oriented operating system saves itself and any child instances," and wherein "overloading a load method in the object-oriented operating system loads itself and any child instances."

Srinvasan is directed to a method of translating relational database tuples to object-oriented objects. While *Srinvasan* describes a method generally similar to that recited by claim 1, *Srinvasan* does not teach, suggest or provide motivation for all of the features claimed by claim 1. *Srinvasan* generally describes a system whereby "[o]bject-oriented applications, such as application 104 to access the data in RDBMS [relational database management system] 112 via an object-relational database gateway 106." (*Srinvasan* at col. 3, lns. 25-28 and FIG. 1). The RDBMS 112 returns tuples 412 (i.e., rows from one or more tables)," and the "object generator receives these tuples 412." (*Id.* at col. 3, lns. 64-67). Then, "the object generator 404 efficiently translates these tuples into object-oriented objects." (*Id.* at col. 4, lns. 3-5).

Srinvasan, however, does not describe many of the features recited by amended claim 1 of the present application. For example, *Srinvasan* does not describe "overloading the load method in the object-oriented operating system to load a latest instance of a table entry." Nor does *Srinvasan* describe "overloading a save method in the object-oriented operating system to save an instance of a table entry."

Nor does *Srinvasan* describe "overloading a remove method in the object-oriented operating system to remove an instance of a table entry." *Srinvasan* also does not describe "overloading a remove method in the object-oriented operating system removes itself and any child instances." Further, *Srinvasan* does not describe "overloading a save method in the object-oriented operating system saves itself and any child instances," nor "overloading a load method in the object-oriented operating system loads itself and any child instances."

While *Srinvasan* does describe the creation of intermediate objects (*Srinvasan* at col. 11, lns. 50-52) and removing intermediate objects from memory (*Id.* at col. 11, lns. 53-55), such described creation and removal of intermediate objects is not the same as the claimed "overloading a save method in the object-oriented operating system saves itself and any child instances,", nor is it the same as a method wherein "overloading a remove method in the object-oriented operating system removes itself and any child instances." as recited by amended claim 1. Thus, while *Srinvasan* does describe the use of intermediate objects, it does not teach, suggest or provide motivation for the features of a save method that saves itself and any child instances, or a remove method that removes itself and any child instances, as recited by amended claim 1 or the present application.

Also, while *Srinvasan* describes the use of derived classes (*Srinvasan* at col. 11, lns. 62-67), the use of derived classes is not the same as "overloading a save method in the object-oriented operating system saves itself and any child instances," nor is it the same as "overloading a remove method in the object-oriented operating system removes itself and any child instances." as recited by amended claim 1. Thus, while *Srinvasan* does teach, suggest or provide motivation for the use of derived classes, it does not teach or suggest the features of a save method that saves itself and any child instances, or a remove method that removes itself and any child instances.

As described in the specification of the present application as published (*See* US 2004/0230555) at paragraph [0087], by way of the claimed invention, "[a]ny parent that has children has those associations automatically generated." As described above, *Srinvasan* does teach, suggest or provide motivation for these beneficial features.

Moreover, as conceded by the Examiner, *Srinvasan* does not teach or suggest or provide motivation for "*overloading* a save method" that "saves itself and any child instances," nor does it teach or suggest "*overloading* a remove method."

Davis is a JAVA programming guide that describes the process of function overloading, which allows for the creation of two methods with the same name, but different arguments. *Davis,* either alone, or in combination with *Srinvasan*, does not teach, suggest or provide motivation for, "representing a relational datababe table as a object-oriented operating system" and "overloading a save method in the object-oriented operating system saves itself and any child instances," or "overloading a remove method in the object-oriented operating system removes itself and any child instances." as recited by amended claim 1.

Accordingly, Applicants submit that neither *Srinvasan* nor *Davis*, either taken alone, or in combination, teaches, suggests, or provides motivation for the combination of features recited by amended claim 1 of the present application.

Claims 6, 9, 10 and 13 depend from, and further narrow and define, claim 1, that has been discussed above and is believed to be allowable over any *Srinvasan-Davis* combination. Accordingly, for at least these reasons, claims 6, 9, 10 and 13 are deemed to distinguish patentably over any hypothetical *Srinvasan-Davis* combination.

Claims 7, 8, 11, 12 and 15 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over *Srinvasan* and *Davis* in view of U.S. Patent No. 5,937,409 (*Wetherbee*).

Claims 7, 8, 11, 12 and 15 depend from, and further narrow and define, claim 1, that has been discussed above and is believed to be allowable over any *Srinvasan-Davis* combination.

Wetherbee is directed to a method for integrating relational databases in an object oriented environment, but does not make up for the deficiencies of *Srinvasan-Davis*.

Accordingly, for at least these reasons, claims 7, 8, 11, 12 and 15 are deemed to distinguish patentably over any hypothetical *Srinvasan-Davis-Wetherbee* combination.

Claim 14 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over *Srinvasan* and *Davis* in view of U.S. Patent No. 5,918,225 (*White*) [1].

Claim 14 depends from, and further narrows and defines, claim 1, that has been discussed above and is believed to be allowable over any *Srinvasan-Davis* combination.

White is directed to a SQL-based database system certain indexing methodaology, but does not make up for the deficiencies of *Srinvasan-Davis*. Accordingly, for at least these reasons, claim 14 is deemed to distinguish patentably over any hypothetical *Srinvasan-Davis-White* combination.

Claim 16 stands rejected under 35 U.S.C. § 103(a) as being unpatentable over *Srinvasan* and *Davis* in view of U.S. Patent No. 6,529,913 (*Doig*).

Claim 16 depends from, and further narrows and defines, claim 1, that has been discussed above and is believed to be allowable over any *Srinvasan-Davis* combination.

Doig is directed to a database system having arrays for storing class information, but does not make up for the deficiencies of *Srinvasan-Davis*. Accordingly, for at least these reasons, claim 14 is deemed to distinguish patentably over any hypothetical *Srinvasan-Davis-Doig* combination.

Thus, applicants submit that each of the claims of the present application are patentable over each of the references of record, either taken alone, or in any proposed hypothetical combination. Accordingly, withdrawal of the rejections to the claims is respectfully requested.

---

[1] *White* is referred to as "Wise" in the Office Action.

Conclusion

In view of the above remarks, reconsideration and allowance of the present application is respectfully requested.

Respectfully submitted,

[illegible]

By: James Dobrow
Attorney for Applicants
Registration No. 46,666

Date: 5 December 2006

Mail all correspondence to:

Docket Administrator
Lowenstein Sandler PC
65 Livingston Avenue
Roseland, NJ 07068

Certificate of Transmission Under 37 CFR 1.8

I hereby certify that this correspondence is being facsimile transmitted to the Untied States Patent and Trademark Office, at facsimile number 571-273-8300

on 5 December 2006

James Dobrow, Reg. No. 46,666
(Name of Registered Rep.)

[illegible] 5 December 2006
(Signature and Date)